PolyMorph: Increasing P300 Spelling Efficiency by Selection Matrix Polymorphism and Sentence-Based Predictions

Alberto Casagrande*[1], Joanna Jarmolowska†[2], Marcello Turconi‡[2], Pierpaolo Busan§[2], Francesco Fabris¶[1], and Piero Paolo Battaglini‖[2]

[1] *Dept. of Mathematics and Geosciences, University of Trieste, Via Valerio 12/1, 34127 Trieste, Italy*

[2] *Dept. of Life Sciences, University of Trieste, Via Weiss 2, 34128 Trieste, Italy*

Abstract

P300 is an electric signal emitted by brain about 300 milliseconds after a rare, but relevant-for-the-user event. One of the applications of this signal is sentence spelling that enables subjects who lost the control of their motor pathways to communicate by selecting characters in a matrix containing all the alphabet symbols. Although this technology has made considerable progress in the last years, it still suffers from both low communication rate and high error rate. This article presents a P300 speller, named PolyMorph, that introduces two major novelties in the field: the selection matrix polymorphism, that reduces the size of the selection matrix itself by removing useless symbols, and sentence-based predictions, that exploit all the spelt characters of a sentence to determine the probability of a word. In order to measure the effectiveness of the presented speller, we describe two sets of tests: the first one *in vivo* and the second one *in silico*. The results of these experiments suggest that the use of PolyMorph in place of the naïve character-by-character speller both increases the number of spelt characters per time unit and reduces the error rate.

*Electronic address: `acasagrande@units.it`; Corresponding author
†Electronic address: `fabasia@libero.it`
‡Electronic address: `marcello.turconi@libero.it`
§Electronic address: `pbusan@units.it`
¶Electronic address: `ffabris@units.it`
‖Electronic address: `battagli@units.it`

1 Introduction

Brain computer interface (BCI) technology allows individuals with motor disabilities to establish a new channel of non-muscular communication with the surrounding environment. Virtual keyboards ruled by computers that collect and interpret user brain signals have been investigated since the rise of this technology. The cerebral activity piloting such keyboards can be revealed with non-invasive methods such as electroencephalography (EEG) [33, 35]. The most frequently studied applications for BCI use the P300 component of the event-related potential (ERP) that manifests following a subject response to an external stimulus [31, 9, 8]. The P300 wave has a positive potential (>10 μV) that appears only after the presentation of an expected or rare stimulus, and has a characteristic distribution in posterior EEG signals (centro-parieto-occipital) [30]. The row-column (RC) speller proposed by Farwell and Donchin in 1988 uses the P300 wave and allows for sequential selection of a character within a matrix of rows and columns [9]. Guger *et al.* reported that the 89% of subjects were able to obtain an accuracy of 80-100% with the P3Speller [12]. However, the RC method has several limitations, one of which concerns the interface itself, which is very tiring for the user, causing a rapid decline in performance. Moreover, selection of a single target can require a relatively long time: 3-8 selections per minute can be obtained with a P300-based BCI [24]. Thus, compared to the 150 words per minute that are produced using normal speech [19], communication through a BCI is much slower and requires significant user attention even to communicate a simple message. Considering the need to increase the efficacy of communication using a BCI, several groups have attempted to improve its performance with P300-based applications that reduce the number of errors while increasing its velocity. Some of these studies have focused on the P300 speller paradigm [28, 32, 14], with the aim of enhancing the classification of the signal [17] to increase the velocity of communication. Salvaris and Sepulveda investigated the effects of visual modification of the P300 speller BCI paradigm by introducing differences in background color, size and style of symbols, and size of background on the display [25]. In that study, although no single visual protocol was best for all subjects, performance could nonetheless be improved using a white background visual protocol; the worst performance was seen with small symbol size. The group of Allison studied the effects of different matrix sizes on P300 amplitude, accuracy, and performance [2]. The results indicated that larger matrices evoked a larger P300 amplitude than a smaller matrix. Among studies that have focused primarily on reducing errors, the adjacency problem, stating that some involuntary selections depend on the distance between characters, has been investigated [11]. In this context, a paradigm based on regions was proposed [10]. This paradigm, thanks to its graphic interface, minimized the effects of overcrowding of stimuli and the adjacency problem. More recently, a predictive spelling system has been introduced. Ryan *et al.*, for example, studied a predictive spelling program (PS) in which a classic RC paradigm was integrated with suggestions based on prefixes of a particular word [24]. An 8×9 matrix was used, and suggestions were

not presented within the selection matrix but in a separate window. This system overcame the non-predictive system in terms of both average time needed to complete a sentence (12 min 43 sec vs 20 min 20 sec, respectively) and average output characters per minute (OCM) ($\mu = 5.28$, $\sigma = 1.67$ vs $\mu = 3.76$, $\sigma = 0.75$) [24]. However, the average accuracy of the predictive system was lower than the non-predictive system (84.88% vs 89.80%, respectively), while there were no significant differences in either bit rates (19.39 vs 17.71, respectively) or selections per minute (3.71 vs 3.76, respectively). Kaufmann *et al.* proposed a different approach which preserved the level of accuracy achieved by the non-predictive speller [15]. In this case, predicted words and alphabetic characters were presented in the selection matrix at the same time. As result, the bit rate (in terms of selections per minute) was high for both the predictive and non-predictive systems (15.7 vs 15.1, respectively). Yet, the predictive system exhibited a higher true bit rate (in terms of characters per minute) with respect to the non-predictive speller (20.6 vs 12, respectively), it required less time to write an entire sentence, and it enhanced the OCM ($\mu = 3.83$, $\sigma = 0.88$ for the predictive speller vs $\mu = 2.12$, $\sigma = 0.52$ for the non-predictive one[1]) [15].

We developed a P300 speller, named *PolyMorph*, with two main aims: increasing the output characters per minute with respect to the speller proposed in the literature and enhancing the spelling accuracy. These goals have been pursued by using classical information theory tools such as *information rate*, *absolute redundancy*, and *adaptive compression*. PolyMorph distinguishes user language from channel code, that is the code used to spell user messages, and it reduces channel code redundancy by transitorily removing symbols that have probability 0 from the selection matrix. Because of this, both the size of selection matrix and the selectable symbols change from selection to selection giving reasons for the speller name. Moreover, PolyMorph adopts a sentence-based prediction system to better model the probability distribution of the user language, improve the word forecasting, and enhance the communication rate.

This work presents Polymorph's features and measures its efficiency by using both *in-vivo* and *in-silico* experiments. The collected results are analyzed and compared to those obtained by the state-of-the-art spellers.

The speller source code, which has been released under the GNU GPL license, and all the data obtained during the experiments are available at URL `http://polymorph.units.it`.

2 PolyMorph

PolyMorph is a P300-based speller that adopts the oddball paradigm to identify one symbol in a (possible) square matrix of potential targets. Polymorph differs from the other spellers on the approach: while most of them try to reduce

[1] The average OCM's are not reported in [15]. We computed them as ratio between the number of characters in the target sentence (i.e., 45) and the overall time needed to spell the sentence (Figure 3 in [15]).

the time required to perform a single selection, Polymorph exploits information theory to reduce the number of selections due to spell a complete sentence.

In order to better understand how PolyMorph works, we first need to introduce in Section 2.1 some notions and the notation that we use along all the paper. In Section 2.2, we will detail PolyMorph features.

2.1 Notions and Notation

In this paper, we adopt the symbol "_" to represent the space character with the aim to reduce ambiguity about the presence or absence of it.

A *string* is a sequence, possibly empty, of symbols in the alphabet $\Sigma = [a - zA - Z.?!\_']$. While we can in theory support accented characters, we decided at the first instance to map them into pairs of characters in Σ to simplify PolyMorph development. For instance, "*È*", "*ó*", "*ä*", and "*ß*" are mapped into "*E'*", "*o'*", "*a*", and "*ss*", respectively.

If s_1 and s_2 are two strings, then $s_1 + s_2$ is the string obtained by concatenating s_2 to s_1. The string s_1 is *suffix* of s_2 if and only if there exists a string s_3 such that $s_2 = s_3 + s_1$; s_1 is *prefix* of s_2 if and only if there exists a string s_3 such that $s_2 = s_1 + s_3$.

A *sentence* is a string whose last symbol is either ".", "?", or "!" and whose shorter prefixes are not sentences, i.e., ".", "?", and "!" can be present only as the last symbol. A *word* is a string in the alphabet $[a - zA - Z']$. Let us notice that the string "*Fermi's*" is considered as a single word. The *suffix word prefix*, or SWP, of a string R is the longest word that is suffix of R. Analogously, the *suffix sentence prefix*, or SSP, of R is the longest sentence that is suffix of R. A string R *completes* a SSP S or a SWP W when there exists a prefix of R that is suffix of S or W, respectively.

Later on, the SWP and the SSP of what has been spelled after n selections from the begin of the session is denoted by W_n and S_n, respectively. Whenever the number of selections are not relevant, we may omit it and write simply W and S. Let us notice that W_n is always a suffix of S_n.

2.2 Features

When a subject uses a brain speller, he is communicating a message (i.e., a sentence) through a channel, whose input is an EEG and the output is a monitor. Since the channel is sequential (i.e., it cannot transfer the entire message in a single step), the message has to be encoded to be transmitted and decoded to be read. We distinguish between *user language* and *channel code* or *code*: the former is the language of the user and it contains all the sentences that may be spelled. The latter is the language of the encoded messages and, in principle, it may account a syntax or an alphabet completely different from that of the user language. Most of the spellers proposed in the literature to date encode the message to be spelled character by character. They also adopt the message alphabet as code alphabet. This code is trivial and, due to the redundancies of natural languages, it may waste time and bit space by transferring unnecessary

symbols. For instance, all the English words that begin by "*xylop*" also begin with "*xylophone*". From this point of view, the suffix "*hone*" is useless, as it brings no further information with respect to "*xylop*", and we can avoid it still preserving the meaning of the message.

Since the '50s, information theory has addressed the problem of reducing the number of bits required to transmit a message and many codes have been proposed in the literature and implemented in real word applications so far [13, 23, 36, 37, 16, 5]. However, these codes are meant to be handled by computer programs and they completely alter the syntactic structure of the messages. On the contrary, we would like to produce a human-oriented code that reduces the number of symbols necessary to spell a sentence in some natural language and, at the same time, minimizes the cognitive overhead required to achieve this goal.

We observed that, in some situations, natural languages have an alphabet that is overabundant with respect to the message chunk to be communicated and, since, the size of an encoded message depends on the cardinality of the code alphabet, this may be a waste of bit space. For instance, none of the English words whose first character is "*k*" have as second one either "*t*" or "*z*". Thus, in some sense, once we know that the first character is "*k*", the symbols "*t*" and "*z*" are unnecessary and they can be transitorily removed from the code alphabet.

A further improvement in the channel code can be obtained by considering word probabilities. Huffman showed that it is possible to minimize the average size of the overall encoded message by representing strings with codes whose length is inversely related to their probability: the higher the probability, the shorted the code [13]. Hence, if we associate some of the selection matrix symbols to word suffices that complete the current SSP with high probability, we may save both useless selections and spelling time.

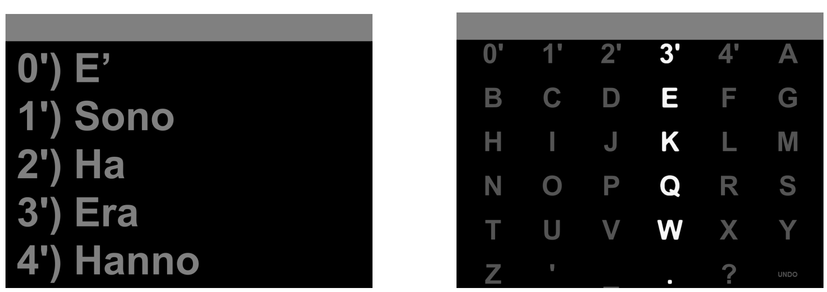
In order to exploit above considerations and improve the efficiency of the channel code, PolyMorph maintains a knowledge-base (KB) that stores all the words that can be spelled, a set of sentences, and the frequencies of both of them. The KB is meant to capture the typical expression and the probability distribution of the language of a single subject. It can be trained at the begin of the session simply by providing a file containing a set of sentences: these sentences, all the words composing them, and their frequencies are automatically evaluated and stored into the KB. During the spelling session, PolyMorph updated the KB every time a sentence is completed and, at the end of the spelling session, it saves the KB in a file that can be reloaded at the beginning the successive session.

The sentences stored in the KB are used to estimate the probability that a word completes the current SSP (i.e., $P(T+R|Q+T)$ where $Q+T$ is the SSP and $T+R$ is the word). For instance, if "*the_word_th*" is the current SSP, its SWP "*th*" is the prefix of the words "*those*", "*the*", and "*that*". However, neither "*the_word_the*" nor "*the_word_those*" appear to be part of an English sentence and there are many chances that none of them have ever been spelled. If this is the case, the KB would suggest that "*that*" has a higher probability than "*those*" and "*the*" to be the next word to be spelled. Since this word probability depends on the SSP, we called it *SSP probability* and we distinguish it from

the *SWP probability* that is the probability to be the next word to be spelled after a SWP. Every time that a sentence is completed, the KB updates both its statistics and the statistics of its words and refines the probability model of the channel code.

At each selection, PolyMorph extracts the SWP W of the current SSP S and it builds the selection matrix that contains three kinds of symbols: 1) *character symbols* (Σ_c), 2) *mandatory symbols* (Σ_m), and 3) *prediction symbols* (Σ_p). Each of the character symbols represents a single character c in the user alphabet. Their number is variable since PolyMorph includes in the selection matrix only those symbols c such that $W + c$ is a prefix for some words in the KB. Because of this, both the size of the spelling matrix and the symbols $\Sigma \stackrel{\text{def}}{=} \Sigma_c \cup \Sigma_m \cup \Sigma_p$ contained in it change from selection to selection (from which the name PolyMorph). Moreover, at the selection of any character symbol c, PolyMorph spells the longest string C such that $W + C$ is a prefix for all the words that begin by $W + c$ in the KB. For instance, if the SWP is “*xylo*” and the character symbol “*p*” is selected, PolyMorph spells “*phone*” because all the words in the KB that begin with “*xylop*” has also “*xylophone*” as prefix. Since the above selection corresponds to browse a radix tree [20] from the node associated to W to the one associated to $W + C$ through an edge whose label is C, we call this feature *label selection*.

The mandatory symbols are always present in the selection matrix and can be configured as required. For instance, they may contain either an undo symbol to delete last selection, punctuation, or a pause symbol to temporarily suspend spelling. In the experimental phase that we present in this paper, we set as mandatory symbols “_” (i.e., the space character), “.”, “?”, and an undo symbol that is meant to cancel last selection.



(a) Prediction phase: the most selected/frequent words are predicted and associated to unique numeric IDs.

(b) Selection phase: the selection matrix is shown and the P300 measurement proceeds.

Figure 1: The working cycle of PolyMorph is split into two phases.

Finally, each of the prediction symbols associates one cell of the selection matrix to a string R+“_” such that $W + R$ is a word in the KB that has either high SSP probability or high SWP probability. In particular, as long as there

are strings for which the SSP probability is defined, those strings are presented, then high SWP probability strings are considered. The number of prediction symbols depends both on KB and on the other symbols in the selection matrix. In order to visually present a –potentially– long string R+"_" in a single cell in the selection matrix, the selection process is split into two phases: the *prediction phase* and the *identification phase* (see Figure 1). The former associate a numeric ID, which is the prediction symbol, to the words $W + R$. The latter exhibits the selection matrix and performs the P300 measurements as done in the row-column paradigm. Whenever a suggesting symbol is selected, the string R+"_" is spelled. For instance, if "*the_word_th*" is the current SSP, then PolyMorph may associate "*that*" to $0'$ –because of the high SSP probability– and "*the*" to $1'$ –because of the high SWP probability. In such this case, if the user selects $0'$, PolyMorph spells "*at_*" and shows the new SSP "*the_word_that_*".

In order to minimize the ratio between the number of selectable symbols (i.e., $|\Sigma|$) and the intensifications required to spell them, both the number of rows (h) and columns (w) of the selection matrix is established dynamically and the system ensures that $h \in [w-1, w]$ holds (i.e., the matrix is almost square). Because of this, $|\Sigma_p|$ may change and, as far as there exist enough words in the KB for which the current SWP is a prefix, it is the smallest natural number, greater than a user parameter $P_{\#}$, such that both $|\Sigma_p| + |\Sigma_c| + |\Sigma_m| = h * w$ and $h \in [w-1, w]$ hold.

A complete description of algorithms and data structures implemented by PolyMorph is given in [4].

3 Metrics

In this section, we present some theoretical notions introduced by information theory to evaluate the redundancy of a language and its compressibility. Moreover, we both described and motivated the metrics used to measure the effectiveness of PolyMorph.

3.1 Information theoretical notions

Let X be a *discrete random variable* with possible values in $\{x_1, \ldots, x_n\}$ and probability P_X. The *entropy* $H(X)$ of X is defined as $H(X) \stackrel{\text{def}}{=} -\sum_{i\in[1,n]} P_X(x_i) * \log P_X(x_i)$.

The *joint entropy* is the entropy of a sequence of random variables, e.g., $H(X_1 \ldots X_n)$. The *information rate*, or simply rate, r is the average entropy per symbol and, in the most general case, it has the form $r \stackrel{\text{def}}{=} \lim_{n\to\infty} H(X_1 \ldots X_n)/n$.

The *maximum entropy* $\hat{H}(X)$ of a random variable X is the greatest entropy achievable by a random variable Y that shares the same support (i.e., the set of values that it can assume) of X. It is defined as:

$$\hat{H}(X) \stackrel{\text{def}}{=} \max_{Y \,\mid\, S(Y)=S(X)} H(Y),$$

where $S(X)$ and $S(Y)$ are the supports of X and Y, respectively.

The *absolute rate* R of a source is the maximum possible rate of information per symbol that can be transmitted by it. If the alphabet Σ of the source does not change during time, it equals $\log|\Sigma|$, while, in the general case, we defined it as $\lim_{n\leftarrow\infty} \hat{H}(X_1 \dots X_n)/n$.

Whenever the random variables are clear from the context, we may write r_n and R_n to denote $H(X_1 \dots X_n)/n$ and $\hat{H}(X_1 \dots X_n)/n$, respectively.

Absolute redundancy estimates how much redundant is a source and, in particular, it evaluates, in average, how many bits per symbol bring no information. More formally, the absolute redundancy D of a language is the difference between its *absolute rate* and its *information rate*, i.e., $D \stackrel{\text{def}}{=} R - r$.

The, so called, *bit rate* estimator [22] has been introduced in the literature to evaluate the maximal flow of information per time unit going through the selector. The original definition is meant to deal with *stationary* and *memoryless* selectors and has the form:

$$B \stackrel{\text{def}}{=} \log_2 n + p \log_2 p + (1-p) \log_2 \frac{1-p}{n-1},$$

where n is the number of selectable objects per selection and p is the probability of correct selection.

3.2 Speller metrics

Bit rate has been successfully used to compare the efficiency of selectors in BCI [34]. However, this metric is strictly related to the size of the selection matrix and it does not take into account the efficiency of communication. In the particular case of spellers, the transfer rate of a set of messages is always upper bounded by the information rate of their language and the average information transmitted by a single selection cannot exceed the information rate of the channel code. Furthermore, even if the maximal rate of a speller A is greater than that of a speller B, A does not necessarily exploit this advantage in transmitting sentences. Because of these reasons, we consider bit rate suitable neither to evaluate nor to compare speller efficiencies.

On the contrary, *absolute redundancy* (AR) can both relate the absolute rate of a speller and the rate of the channel code and measure how many bit are wasted during a single symbol selection. The smaller the absolute redundancy, the more efficient the speller is. The effects of AR on the identification phase were evaluated by the number of *intensifications per selection and repetition* (ISR) that is equal to $I/(S * R)$ where I, S, R are the total number of intensifications, the number of selections necessary to spell the target, and the selection repetitions, respectively. Above metrics record the redundancy of spelling process, but do not take into account the prediction phase. Thus, we also reported both *output characters per minute* (OCM) and the *mean number of selections per minute* (SM) to estimate the PolyMorph communication speed-up.

Finally, the correlation between the use of Polymorph and the number of spelling errors was highlighted by two different metrics: the *accuracy* (AC),

that is number of correct selections divided by the total number of selections, and the *errors per character* (EC), that is the ratio between the number of errors and the length of the spelt sentence.

4 Methods

We performed two kinds of tests: an *in-vivo* set of tests, which aimed at evaluating the performances of PolyMorph on real users, and an *in-silico* set of tests, which unravel the relation between the user's language and PolyMorph effectiveness. In both the cases, we also tested a classic P3Speller [26] to compare the efficiency of the two spellers.

4.1 In-Vivo

4.1.1 Partecipants

The present study considers a total of 10 healthy subjects including 6 males and 4 females. Their ages range from 22 to 29 years with an average of 24.9 years and a standard deviation of 1.9. All of them were Italian native speakers and they were not experts in BCI systems. The experimental protocol was prepared in accordance with the Declaration of Helsinki, and was approved by the local ethics committee. Moreover, all subjects provided signed informed consent forms before the study was initiated.

4.1.2 Experimental Paradigm

Each of the considered subjects underwent an experiment consisting of two parts: a *set-up session*, which included EEG calibration and a learning phase for the P300 identification procedure, and an *on-line session*, which called for some spellings by using both PolyMorph and P3Speller.

The effectiveness of the proposed system strongly depends on its KB and, because of this reason, the on-line session involved both a sentence that was already present in it (*target sentence A*) and a one that was not included into the system at beginning of the experiment (*target sentence B*).

Each sentence was composed twice (*turn 1* and *turn 2*) by using PolyMorph to investigate the aftermath of a spelling on its knowledge-base and, as a consequence, on the following spellings. Since the performances of the standard P3Speller depends neither on a knowledge-base nor on the character distribution of the sentence to be spelled, we only required a single spelling of target sentence A for it. This results in five different spelling sessions, i.e., two spellings of sentence A by using PolyMorph, two spellings of sentence B by using PolyMorph, and one spelling of sentence A by using P3Speller. With the aim of avoiding any bias due to reduced weariness of the subjects, the order of these sessions was pseudo-randomized for all participants.

Since all the subjects involved in our experiments are Italian native speakers, we decided to use Italian language during the experiment to avoid any cognitive

effort other than that required to spell a sentence through P300. While the effectiveness of the PolyMorph system depends on the adopted language, we expect to obtain the very same conclusions for any language that contains some kind of redundancy, as in the case of the natural languages.

4.1.3 PolyMorph Knowledge-Base

In order to initially fill the knowledge-base of PolyMorph, we extracted the 200 most commonly used Italian nouns, adjectives, and verbs from `http://telelinea.free.fr/italien/1000_parole.html`. After that, we collected the 107.075 most common sentences containing these words from the '*Corpus dell'italiano*' of the *Istituto di Linguistica Computazionale* (`http://www.ge.ilc.cnr.it/page.php?ID=moduli_verbi&lingua=it`) and we store them into the system. Finally, we added 4101 sentences by randomly selecting them from the web. In this way, the initial knowledge-base accounted 111,176 sentences and 51,590 words; in average, the sentences were composed by 37 characters and contained 5.3 words.

In the following, this phrasebook is called *Italian curated phrasebook* (**It***).

4.1.4 Target sentences

The sentences "*Piace tanto alla gente.*" ("People like it very much.") and "*Sono andato sulla luna.*" ("I went to the Moon.") were chosen as target sentence A and target sentence B, respectively. Both these sentences consist of 23 characters, including the spaces and period, all their words were contained into the initial knowledge-base (see Sec. 4.1.3).

4.1.5 On-line session

The on-line session consisted of spelling the two target sentences (see Sec. 4.1.4). During this session, both the duration of the stimulus and the pause between two consecutive stimuli (*inter stimulus interval*) were set to 125 ms. The time between the appearance of the selection matrix and the first stimulus (*pre-sequence duration*) was 3 s in both systems (P3Speller and PolyMorph). The time between the last stimulus and the appearance of the selection matrix (*post-sequence duration*) was 3 s in the P3Speller, while the duration of the suggested phrase in the PolyMorph system was set to 10 s.

4.1.6 Data acquisition

The EEG was registered using a standard cap (Electro-Cap International, Inc.) with a modified version of the international placement system to position electrodes. The electrodes used for registration were located in the fronto-centro-parietal-occipital cortex, and in particular the following electrodes were used: Fz, Cz, P3, Pz, P4, PO7, Oz, and PO8. The right mastoid was used as a reference for these electrodes, and the left mastoid was used as the ground.

Impedance was maintained below 5.0 kΩ. The signal was amplified and digitized with a Micromed amplifier (SAM 32FO fc1; Micromed S.p.A., Italy; analog high-pass filter 0.1 Hz; sampling frequency 256 Hz). The signal registered in each channel was processed with a common average reference spatial filter. From each EEG channel a data epoch of 800 ms was extracted after presentation of the stimulus. BCI2000 software was used to manage the experiment [26] (i.e., for presentation of stimuli, collection and elaboration of EEG data, and management of the spelling session).

4.1.7 Set-up session

The set-up session was carried out by using the P3speller software in "copy mode" on a 6×6 matrix. In this session, the subjects were requested to spell five alphanumeric strings that were composed of four symbols each. The strings were presented on the upper left side of the computer monitor and each successive symbol to be selected was emphasized by surrounding it by parentheses. Both the duration of each stimulus and the duration of the inter-stimulus interval were 125 ms. Illuminations were organized in sequences (in a pseudo-randomized order) of row-column illuminations in which each row and column was illuminated only once. A total of 14 sequences were set for each individual symbol. In this way, a total of 28 illuminations were obtained for the target stimulus (the number of illuminations for each row plus the number of illuminations for each column) and another 140 illuminations for non-target stimuli. Each item was selected for 42 s, while the duration of the block lasted about 3 minutes. Considering the pre/post run pause and the pre/post stimulus pause, the entire initial session lasted about 15 minutes.

4.1.8 P300 identification

The identification of the P300 component was performed during presentation of the target stimulus. The tool "P300 Classifier" that is incorporated in the BCI2000 software was used. Stepwise linear discriminant analysis (SWLDA) was used in this phase of the experiment. This method assumes that the P300 is obtained for one of the six row/column intensifications, and that the P300 response is not varied compared to the row/column stimuli. The resulting classification was taken as the maximum of the sum of the characteristic vectors obtained for the respective rows and columns [18]. As a result of the process of discrimination, a transition matrix was generated that estimates the probability of the system obtaining a definitive answer (in terms of P300) for each participant. The procedure allows choosing the optimal number of repetitions per stimulus (NRS).

4.1.9 On-line parameters

Both the *stimulus duration* (SD) and the *inter-stimuli interval* (ISI) were set to 125ms. The *pre-sequence duration* (PreS), i.e., the time between the appearance

of the selection matrix and the first stimulus, was set to 3s for both P3Speller and PolyMorph. The *post-sequence duration* (PostS), i.e., the time between the last stimulus and the change of selection matrix in P3Speller, was set to 3s, while the prediction phase lasts 10s (PPD). For each subject, we set the minimal NRS that guarantees to maintain the accuracy of the set-up session to 100%: from the first to the tenth subject, NRS was set to 6, 14, 12, 20, 13, 6, 9, 11, 14, and 11, respectively.

4.2 In-Silico

The efficiency of PolyMorph depends on the language adopted by users: the more redundant it is, the more efficient PolyMorph is expected to be. *In-silico* experiments aimed at unravelling this dependency.

4.2.1 Experimental paradigm

We decided to focus on languages based on Latin alphabet and, in particular, on English (**En**) and German (**Ge**), belonging to the Germanic branch of the Indo-European language family, French (**Fr**) and Italian (**It**), of the Italic branch of the same family, and Finnish (**Fi**) and Hungarian (**Hu**), two representatives of the Uralic languages. The Italian curated phrasebook (**It***) was also considered to better compare *in-vivo* and *in-silico* experimental results.

The first analysis estimated how much redundant is the channel code of both P3Speller and PolyMorph varying the user language and it provided a measure of the spelling efforts wasted due to redundancy. Since the channel code of PolyMorph depends exclusively on the KB, we were required to build a phrasebook $\mathcal{P}_{\mathcal{L}}$, to store all its sentences in a language-specific KB, named $\mathrm{KB}_{\mathcal{L}}$, and, finally, to infer AR, for each of the considered user languages $\mathcal{L}$.

As it concerns the second part of the *in-silico* analysis, we investigated how the efficiency of PolyMorph and P3Speller is related to both user language and target sentences. For each language $\mathcal{L}$, we built two phrasebooks: $I(\mathcal{L})$, whose sentences were all included into $\mathcal{P}_{\mathcal{L}}$, and $O(\mathcal{L})$, whose sentences were not included into $\mathcal{P}_{\mathcal{L}}$. We simulated the behaviour of PolyMorph during the spelling of the sentences in $I(\mathcal{L})$ and $O(\mathcal{L})$ by using $\mathrm{KB}_{\mathcal{L}}$ and we recorded the sizes of the spelling matrix for all the selections.

Above experiments were repeated avoiding both the prediction phase and all the prediction symbols with the intent of better understand the relevance of each of the PolyMorph features.

4.2.2 Phrasebooks and knowledge-bases

We wrote a computer program to collect sentences from the web and we built, for each of the considered languages $\mathcal{L}$, a phrasebook $\mathcal{T}_{\mathcal{L}}$ containing 100,000 –possibly repeated– sentences. Any character having an accent, was replaced by the corresponding ASCII character and, if it was the last of a word, we added an

apostrophe after it. Moreover, all the characters that are not in $[a-zA-Z.?]$ were removed.

The phrasebooks $I(\mathcal{L})$ and $O(\mathcal{L})$ are subsets of $\mathcal{T}_{\mathcal{L}}$ and each of them contains 10,000 distinct sentences. The sentences were included either in $I(\mathcal{L})$ or in $O(\mathcal{L})$ by a computer program. The same program built the phrasebook $\mathcal{P}_{\mathcal{L}}$ as the phrasebook that contains all the sentences in $\mathcal{T}_{\mathcal{L}}$, but those in $O(\mathcal{L})$. The knowledge-base $\mathrm{KB}_{\mathcal{L}}$ was filled by using all the sentences in $\mathcal{P}_{\mathcal{L}}$ and all the words in $O(\mathcal{L})$ that are not contained in $\mathcal{P}_{\mathcal{L}}$.

	$O(\mathcal{L})$				$I(\mathcal{L})$			
$\mathcal{L}$	Characters	Words	Sentences	C/W	Characters	Words	Sentences	C/W
En	1,104,177	180,161	10,000	6.13	1,166,867	189,938	10,000	6.14
Ge	1,015,970	137,598	10,000	7.38	1,021,348	137,450	10,000	7.43
Fr	1,183,358	185,089	10,000	6.39	1,055,790	166,292	10,000	6.35
It	1,132,476	172,385	10,000	6.57	1,011,753	154,604	10,000	6.54
Fi	898,813	102,291	10,000	8.79	875,585	99,874	10,000	8.77
Hu	914,398	146,779	10,000	6.23	953,824	151,024	10,000	6.32
It*	356,238	55,005	10,000	6.48	383,541	59,631	10,000	6.43

Table 1: $O(\mathcal{L})$'s and $I(\mathcal{L})$'s statistics. The column C/W reports the average characters per word.

4.2.3 Absolute redundancy evaluation

Absolute redundancy computation requires to evaluate the entropy of the channel code. Unfortunately, accurate estimation of the entropy for natural language is a complex task that requires thousands of experiments (see [29, 27, 3]) and, for sure, it goes far beyond the intent of this work. Because of this, we decided to approximate its evaluation both by assuming independence between two successive words of the user language and by restricting the words to the ones contained in a dataset.

The absolute rate of the channel code was lower bounded by $\log|\Sigma|$ (i.e., $\approx$ 4.95) for P3Speller and approximated by R_{1000} for PolyMorph. In the same way, the rates of the channel code of both PolyMorph and P3Speller were estimated as r_{1000} and their absolute redundancies were appraised as $D_{1000} \stackrel{\text{def}}{=} R_{1000} - r_{1000}$ and $\overline{D}_{1000} \stackrel{\text{def}}{=} R - r_{1000}$, respectively.

4.2.4 Simulation parameters and other metrics computation

The NRS was set to the mean of the user NRS's of the *in-vivo* experiments (i.e., 12). As concern the remaining parameters, they were left unchanged with respect to whose reported in Section 4.1.9.

By taking into account this setting and the size of the selection matrix at each selection, which was obtained as output from the simulations, we were able to compute ISR, OCM, and SM. In particular, the number of intensifications

required by both PolyMorph and P3Speller to spell one symbol on a $h \times w$ speller matrix is $N_i = (h + w) * \mathrm{NRS}$ and the time spent in such a selection is $N_i * \mathrm{SD} + (N_i - 1) * \mathrm{ISI} + \mathrm{PPD} + \mathrm{PreD}$ for PolyMorph and $N_i * \mathrm{SD} + (N_i - 1) * \mathrm{ISI} + \mathrm{PostD} + \mathrm{PreD}$ for P3Speller.

Since the *in-silico* experiments were carried out automatically, they did not contain spelling errors and, thus, we did evaluate neither AC nor EC for them.

5 Data Analysis

5.1 Dependent variables

In order to understand the relevance of the data coming from both *in-vivo* and *in-silico* experiments, ISR, OCM, SM, AC, and EC were used as dependent variables in repeated-measures analysis of variance (rmANOVA) when data were normally distributed (data normality was verified by using Shapiro-Wilk test). First, the factors included in the model were related to sentence writing (sentence A) and speller system (three levels: turn 1 and 2 of PolyMorph and then P3Speller). Moreover, we performed two-way rmANOVA, in this case factors included in the model were related to sentence writing (two levels: sentence A and sentence B) and speller system (two levels: turn 1 and 2 for PolyMorph). A p-value level of < 0.05 was considered statistically significant. In cases where a significant interaction and/or when an effect representing a main factor was detected, post-hoc analysis was carried out using a Student's t-test.

When not normally distributed data were present, we used non-parametric methods. In particular, the Friedman test was used for one-way repeated measures analysis of variance by ranks. In this case, post-hoc analysis was performed by using Wilcoxon Signed Rank Test. Statistical tests are always intended as two-tailed.

6 Results

In the remaining parts of this article, we write μ, σ, and ρ to denote means, standard deviations, and correlations, respectively.

6.1 In-vivo

Concerning sentence A, we observed larger OCM in both turn 1 and 2 of PolyMorph session with respect to that obtained by using P3Speller ($df = 2$; $\chi^2 = 20$; p-value < 0.00005; Wilcoxon test: p-value $= 0.00195$ in both cases). It was also detected relevant enhancement of OCM from turn 1 to turn 2 (Wilcoxon test: p-value $= 0.00195$).

The rmANOVA showed a significant interaction between writing system and spelled sentence ($F_{(1;9)} = 6.33$; p-value $= 0.033$). With regard to PolyMorph, we found a substantially increased OCM in turn 1 of sentence A compared with the OCM of the same turn of sentence B ($t_{(9)} = 6.69$; p-value < 0.0009).

Finally, the OCM obtained during turn 2 is larger than that of turn 1 for both sentence A and B ($t_{(9)} = 4.2$; p-value < 0.002 and $t_{(9)} = 9.88$; p-value < 0.0009, respectively).

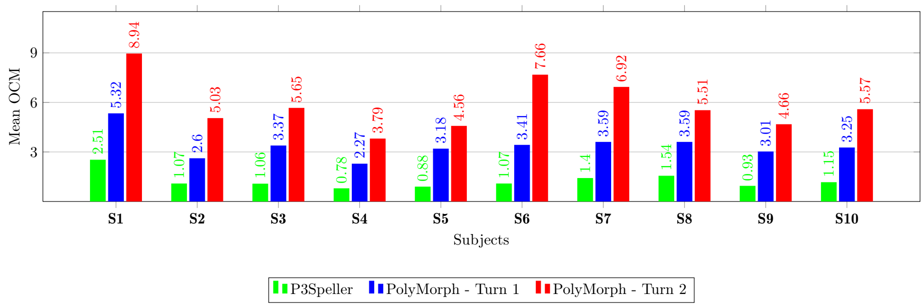


Figure 2: Mean output characters per minute *in-vivo*: the values reported for PolyMorph turn 1 and 2 are the means of the spelling times of sentence A and B in turn 1 and turn 2 respectively.

When accuracy was assessed, there were significant differences among conditions ($df = 2$; $\chi^2 = 6.87$, p-value $= 0.032$). Post-hoc analysis show a significant enhancement of accuracy when spelling the sentence A for the 1$^{\text{st}}$ time (turn 1) by PolyMorph (and, partially, also for turn 2) with respect to that obtained by P3Speller (Wilcoxon test: p-value $= 0.039$ and p-value $= 0.074$, respectively).

	Sentence A			Sentence B	
Subj	PM 1	PM 2	P3S	PM 1	PM 2
S1	7/7	5/5	23/23	11/11	5/5
S2	8/9	5/5	25/27	11/11	5/5
S3	7/7	5/5	27/31	11/11	5/5
S4	7/7	5/5	25/27	11/11	5/5
S5	7/7	5/5	29/35	11/11	6/7
S6	10/12	5/5	37/54	13/15	6/7
S7	8/9	5/5	26/29	11/11	5/5
S8	7/7	6/6	23/23	11/11	5/5
S9	7/7	6/6	27/31	11/11	5/5
S10	7/7	5/5	27/31	13/15	6/6
Total	75/79	52/52	272/311	114/118	53/55

(a) Accuracy (AC). Data are reported in the format "(number of correct selections)/(total number of selections)".

	Sentence A			Sentence B	
Subj	PM 1	PM 2	P3S	PM 1	PM 2
S1	0/23	0/23	0/23	0/23	0/23
S2	1/23	0/23	2/23	0/23	0/23
S3	0/23	0/23	4/23	0/23	0/23
S4	0/23	0/23	2/23	0/23	0/23
S5	0/23	0/23	6/23	0/23	1/23
S6	2/23	0/23	17/23	2/23	1/23
S7	1/23	0/23	3/23	0/23	0/23
S8	0/23	0/23	0/23	0/23	0/23
S9	0/23	0/23	4/23	0/23	0/23
S10	0/23	0/23	4/23	2/23	0/23
Total	4/230	0/230	41/230	4/230	2/230

(b) Errors per character (EC). Data are reported in the format "(wrong selections)/(number of characters in the sentence)".

Table 2: Accuracy (AC) and errors per character (EC) for PolyMorph turn 1 (PM 1), turn 2 (PM 2), and P3Speller (P3S) for *in-vivo* experiments.

Errors per each character selected were also assessed. Test indicates that the distribution of results significantly differ among conditions ($df = 2$; $\chi^2 = 14.89$, p-value $= 0.00058$). Post-hoc analysis exhibits a significant reduction of errors per each character in both PolyMorph turn 1 and 2 of sentence A with respect to

errors per each character obtained when using P3Speller (Wilcoxon test: p-value is 0.0078 in both cases).

6.2 In-silico

PolyMorph reduces the absolute redundancy of the channel code in a significant way with respect to P3Speller for all the considered languages ($\mu \approx 0.93$, $\sigma \approx 0.11$ smaller than $\mu \approx 4.13$, $\sigma \approx 0.13$, Wilcoxon test: p-value = 0.015626), but the amplitude of this reduction does not depend in a substantial way on the adoption of a specific language. The AR reduction affects ISR which is also decreased. However, due to the prediction characters introduced in PolyMorph, this contraction becomes evident exclusively when the sentence to be spelt is not present in the KB (see Tables 6). As the matter of facts, Wilcoxon test revealed a significant reduction in ISR when using PolyMorph in place of P3Speller in such a case ($\mu \approx 11.22$, $\sigma \approx 0.37$ smaller than 12.00, Wilcoxon test: p-value = 0.015626). On contrary, the same test does not highlight differences between PolyMorph and P3Speller in ISR when the spelt sentence is contained in the KB ($\mu \approx 12.42$, $\sigma \approx 0.57$ vs 12.00, Wilcoxon test: p-value = 0.09375), but it tops again P3Speller if we take into account the additional row due to prediction (13.00 for P3Speller with an additional row, Wilcoxon test: p-value = 0.015626).

	P3Speller			PolyMorph - Pred.			PolyMorph - No Pred.		
$\mathcal{L}$	$R>$	r_{1000}	$\overline{D}_{1000}$	R_{1000}	r_{1000}	D_{1000}	R_{1000}	r_{1000}	D_{1000}
En	4.95	0.97	3.98	4.85	4.01	0.84	4.85	4.30	0.29
Ge	4.95	0.80	4.15	4.82	3.91	0.91	4.82	4.23	0.32
Fr	4.95	0.85	4.10	4.82	3.95	0.87	4.82	4.26	0.31
It	4.95	0.81	4.14	4.81	3.99	0.83	4.81	4.26	0.28
Fi	4.95	0.73	4.22	4.79	3.82	0.96	4.79	4.21	0.39
Hu	4.95	0.63	4.32	4.77	3.81	0.97	4.77	4.19	0.38
It*	4.95	0.98	3.97	4.73	3.60	1.13	4.73	4.13	0.52

Table 3: Channel code entropies and absolute redundancy. By removing predictions, we could reduce the absolute redundancy of the PolyMorph channel code, however, this choice would reduce the speller efficiency.

PolyMorph exhibits an increased OCM with respect to P3Speller whether the spelt sentences are contained into the KB ($\mu \approx 6.38$, $\sigma \approx 0.81$ greater than 1.43, Wilcoxon test: p-value = 0.015626) or not ($\mu \approx 2.86$, $\sigma \approx 0.35$ greater than 1.43, Wilcoxon test: p-value = 0.015626). With no doubt, the former case is the most favorable one as it causes an average improvement of about 350% with respect to P3Speller, but the latter case has a still remarkable mean enhancement of about 100%. The amplitude of the improvement depends, as expected, on the user language with peaks of about 430% for Finnish $I(\mathcal{L})$ (former case) and 150% for spelling Hungarian $O(\mathcal{L})$ (latter case) and there is a high correlation between the measured OCM and the average characters per

word ($\rho = 0.76$ and, if we do not consider **It***, $\rho = 0.98$). Intriguingly, there is no apparent relation between the OCM gain in spelling either $I(\mathcal{L})$'s or $O(\mathcal{L})$'s (i.e., sentences in the KB and sentences outside the KB) ($\rho \approx 0.04$ and, if we do not consider **It***, $\rho \approx -0.23$). As it concerns the former condition test set, the worst case language appears to be the one of the Italian curated phrasebook. This is due to the sentence lengths in the phrasebooks: since the sentences in the Italian curated phrasebook are shorter on average than those in the other phrasebooks (see Table 1), the SSP based prediction decreases its effectiveness.

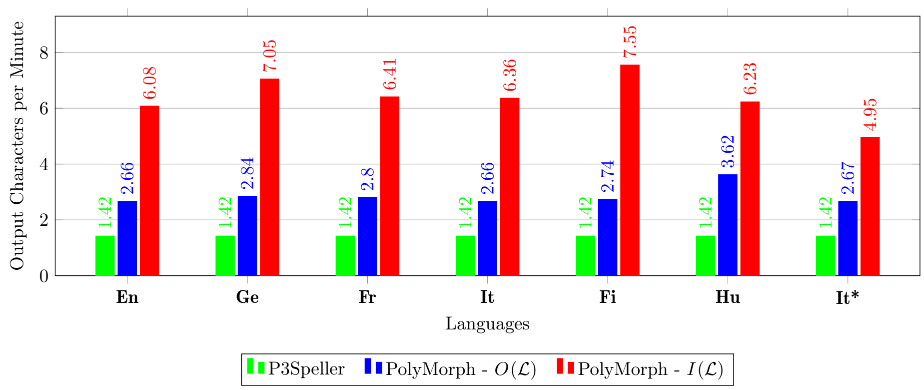


Figure 3: Mean output characters per minute for *in-silico* experiments when predictions are enabled.

The data measured when predictions were disabled highlighted that most of the PolyMorph speed-up is due to the predictions themselves (see Table 7). However, label selections and matrix polymorphism alone significantly increased OCM with respect to P3Speller in spelling both $O(\mathcal{L})$ ($\mu \approx 1.88$, $\sigma \approx 0.09$ greater than 1.43, Wilcoxon test: p-value $= 0.015626$) and $I(\mathcal{L})$ ($\mu \approx 1.88$, $\sigma \approx 0.10$ greater than 1.43, Wilcoxon test: p-value $= 0.015626$).

While the use of PolyMorph in place of P3Speller increases OCM, SM drop from 1.43 to $\mu \approx 1.29$ ($\sigma \approx 0.03$), in spelling sentences that are not contained in the KB (Wilcoxon test: p-value $= 0.015626$), and to $\mu \approx 1.20$ ($\sigma \approx 0.04$), in spelling sentences that are in the KB (Wilcoxon test: p-value $= 0.015626$). In both the cases, SM and ISR have a high correlation: -0.995336 and -0.998081 in spelling $O(\mathcal{L})$'s and $I(\mathcal{L})$'s, respectively.

7 Conclusions and Future Works

This paper presents a P300-based speller, named PolyMorph, which, among the other features, aims at minimizing the size of the selection matrix and predicts the next word to be spelt taking into account what have been already selected. While word prediction has been already proposed in the literature (e.g., see [24, 6, 15]), as far as we known, sentence-based predictions, polymorphic

speller matrix, and label selection are exclusive features in the field so far. In order to implement the last two features, PolyMorph assumes to have a complete knowledge of the user's dictionary. We do not consider this constraint particularly restrictive, however, we plan to remove it in the future.

PolyMorph differs from the other spellers also because it splits each working cycle into a suggestion phase and an identification phase. If, from one hand, this choice increases the time due to perform a selection, from the other hand, it allows us to have symbols of similar size in the selection matrix.

We carried out some *in-vivo* and *in-silico* experiments. Although these tests are limited in number and do not guarantee the same results in subjects with disabilities, they furnish a cheering picture and push us to further investigate PolyMorph. In particular, they prove, as already noticed in [24] and [15], that predictive text entries can increase spelling efficiency. They also highlight that sentence-based predictions play a crucial role in the OCM improvements made by PolyMorph (see Section 6). While the OCM measured by spelling for the first time a sentence that is not present in the KB (target sentence B, turn 1) is not really impressive with respect to the ones obtained in [24] and [15] ($\mu \approx 2.70$, $\sigma \approx 0.67$ vs $\mu \approx 5.28$, $\sigma \approx 1.67$ and $\mu \approx 3.83$, $\sigma \approx 0.88$, respectively[2]), targeting a sentence that is already contained in the KB (target sentence A, turn 1) or, better, a sentence that is "frequently" spelt by the subject (both target sentences A and B, turn 2) produces a massive breakthrough in OCM ($\mu \approx 4.02$, $\sigma \approx 1.03$, $\mu \approx 5.99$, $\sigma \approx 1.78$, and $\mu \approx 5.76$, $\sigma \approx 1.52$, respectively). Obviously, a proper comparison of these results should take into account many factors, such as the average size of the spelt words or the length of the spelt sentence. However, these data appear to be more significant at the light of the SM's that, in average, were $\mu \approx 3.71$, $\sigma \approx 0.75$ and $\mu \approx 2.12$, $\sigma \approx 0.52$ in the works of Ryan *et al.* [24] and Kaufmann *et al.* [15], respectively[3], and $\mu \approx 1.35$, $\sigma \approx 0.35$ for all the PolyMorph sessions i.e., turns 1 and 2 for both target sentences A and B. This means that, when the sentence to be spelt is contained in the KB, the OCM's obtained by using PolyMorph are comparable to those measured in [24] and [15] (target sentence B, turn 1) and, in some cases, higher (target sentences A and B, turn 2) even if the proposed speller maintains SM's to a fraction of those of its competitors (about the 35% of those indicated by Ryan *et al.* [24] and about 62% of those estimated from the work of Kaufmann *et al.* [15]).

Another major result obtained by PolyMorph concerns user accuracy. In contrast to the spellers presented in [24] and [15] which either reduce it (the former) or preserve it (the latter) with respect to P3Speller, PolyMorph increased AC from 272/311 on P3Speller (the average probability of correct selection is $\mu \approx 0.89$, $\sigma \approx 0.09$) to 75/79 ($\mu \approx 0.96$, $\sigma \approx 0.06$), 52/52 ($\mu = 1.00$, $\sigma = 0.00$),

[2] The average OCM's are not reported in [15]. We computed them as ratio between the number of characters in the target sentence (i.e., 45) and the overall time needed to spell the sentence (Figure 3 in [15]).

[3] The average SM's are not reported in [15]. Since the authors stated that there is no significant differences between predictive and non-predictive speller about SM, we approximate them as the SM for non-predictive speller, i.e., the ratio between the number of characters in the target sentence (i.e., 45) and the overall time needed to spell the sentence (Figure 3 in [15]).

114/118 ($\mu \approx 0.97$, $\sigma \approx 0.06$), and 53/55 ($\mu \approx 0.97$, $\sigma \approx 0.06$) in spelling target sentence A, turn 1 and 2, and target sentence B, turn 1 and 2, respectively. This trend was confirmed also by EC which went from 41/230 (the probability of wrong selection per character is $\mu \approx 0.18$, $\sigma \approx 0.21$) to 4/230 ($\mu \approx 0.02$, $\sigma \approx 0.03$), 0/230 ($\mu \approx 0.00$, $\sigma \approx 0.00$), 4/230 ($\mu \approx 0.02$, $\sigma \approx 0.04$), and 2/230 ($\mu \approx 0.01$, $\sigma \approx 0.02$), respectively.

The reported results are exciting, but they strictly depend on the sentence to be spelt and, in particular, on its relative frequency in the subject language. Since the subject language itself is user dependent and it has many chances to be affected by the spelling device (e.g., see the case of SMS [1, 21, 7]), we plan to perform long-term experiments to evaluate the impact of the proposed techniques on a real spelling environment.

We observed that almost no error had been accounted during PolyMorph sessions. Thus, in the future, we will investigate the relation between time parameters, output characters per minute, and accuracy. We will also remove some of the constraints of the current version, for instance, by allowing users to dynamically enrich the vocabulary. Finally, we would like to integrate in PolyMorph a knowledge graph that provides the most likely sentences according to the situation. Such a context-aware mechanism will increase the odd of predict the word that is going to be spelt by the user and, as a consequence, will have positive effects on the OCM.

Acknowledgements

This work has been partially supported by Istituto Nazionale di Alta Matematica (INdAM) and by University of Trieste as one of the outcomes of the project FRA 2014 "Learning specifications and robustness in signal analysis (with a case study related to health care)".

References

[1] Sabreena Ahmed, Abu Sadat Nurullah, and Subarna Sakar. The Use of SMS and Language Transformation in Bangladesh. *Spectrum*, 6&7:107–139, 2010.

[2] Brendan Z Allison and Jaime A Pineda. ERPs evoked by different matrix sizes: implications for a brain computer interface (BCI) system. *IEEE Trans Neural Syst Rehabil Eng*, 11(2):110–113, Jun 2003.

[3] F. Behr, V. Fossum, M. Mitzenmacher, and D. Xiao. Estimating and comparing entropy across written natural languages using ppm compression. In *Proceedings of the Data Compression Conference (DCC '03)*, pages 416–425. IEEE, 2003.

[4] A. Casagrande, J. Jarmolowska, M. Turconi, F. Fabris, and Battaglini P.P. PolyMorph: A P300 Polymorphic Speller. In *Proceedings of The Interna-*

tional Conference on Brain & Health Informatics (BHI'13), volume 8211 of *lncs*, pages 297–306. Springer, October 2013.

[5] John G. Cleary and I. Witten. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, 32(4):396–402, Apr 1984.

[6] T. D'Albis, R. Blatt, R. Tedesco, L. Sbattella, and M. Matteucci. A predictive speller controlled by a brain-computer interface based on motor imagery. *ACM Trans. Comput.-Hum. Interact.*, 19(3):20:1–20:25, October 2012.

[7] Solomon Ali Dansieh. SMS Texting and Its Potential Impacts on Students' Written. *International Journal of English Linguistics*, 1(2):222–229, 2011.

[8] Connie C Duncan, Robert J Barry, John F Connolly, Catherine Fischer, Patricia T Michie, Risto Näätänen, John Polich, Ivar Reinvang, and Cyma Van Petten. Event-related potentials in clinical research: guidelines for eliciting, recording, and quantifying mismatch negativity, P300, and N400. *Clin Neurophysiol*, 120(11):1883–1908, Nov 2009.

[9] L. A. Farwell and E. Donchin. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalogr Clin Neurophysiol*, 70(6):510–523, Dec 1988.

[10] Reza Fazel-Rezai and Kamyar Abhari. A comparison between a matrix-based and a region-based P300 speller paradigms for brain-computer interface. *Conf Proc IEEE Eng Med Biol Soc*, 2008:1147–1150, 2008.

[11] Reza Fazel-Rezai and Waqas Ahmad. P300-based brain-computer interface paradigm design. *Recent Advances in Brain-Computer Interface Systems*, 2011.

[12] Christoph Guger, Shahab Daban, Eric Sellers, Clemens Holzner, Gunther Krausz, Roberta Carabalona, Furio Gramatica, and Guenter Edlinger. How many people are able to control a P300-based brain-computer interface (BCI)? *Neurosci Lett*, 462(1):94–98, Oct 2009.

[13] David Huffman. A method for the construction of minimum redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101, 1952.

[14] T Kaufmann, S M Schulz, C Grünzinger, and A Kübler. Flashing characters with famous faces improves ERP-based brain-computer interface performance. *J Neural Eng*, 8(5):056016, Oct 2011.

[15] Tobias Kaufmann, Stefan Völker, Laura Gunesch, and Andrea Kübler. Spelling is Just a Click Away - A User-Centered Brain-Computer Interface Including Auto-Calibration and Predictive Text Entry. *Front Neurosci*, 6:72, 2012.

[16] J.C. Kieffer and En hui Yang. Grammar-based codes: a new class of universal lossless source codes. *IEEE Transactions on Information Theory*, 46(3):737–754, May 2000.

[17] D J Krusienski, E W Sellers, D J McFarland, T M Vaughan, and J R Wolpaw. Toward enhanced P300 speller performance. *J Neurosci Methods*, 167(1):15–21, Jan 2008.

[18] Dean J Krusienski, Eric W Sellers, François Cabestaing, Sabri Bayoudh, Dennis J McFarland, Theresa M Vaughan, and Jonathan R Wolpaw. A comparison of classification techniques for the P300 Speller. *J Neural Eng*, 3(4):299–305, Dec 2006.

[19] Howard Maclay and Charles Osgood. Hesitation Phenomena in Spontaneous English Speech. *Word*, 15:19–44, 1959.

[20] D. R. Morrison. PATRICIA - Practical Algorithm To Retrieve Information Coded in Alphanumeric. *J. ACM*, 15(4):514–534, 1968.

[21] Nancy Anashia Ong'onda, Peter Maina Matu, and Pamela Anyango Oloo. Syntactic aspects in text messaging. *World Journal of English Language*, 1(1):2–8, 2011.

[22] J.R. Pierce. *An Introduction to Information Theory: Symbols, Signals & Noise*. Dover Books on Mathematics Series. Dover Publications, 1980.

[23] J.J. Rissanen. Generalized kraft inequality and arithmetic coding. *IBM Journal of Research and Development*, 20(3):198–203, May 1976.

[24] D. B. Ryan, G. E. Frye, G. Townsend, D. R. Berry, S. Mesa-G, N. A. Gates, and E. W. Sellers. Predictive spelling with a P300-based brain-computer interface: Increasing the rate of communication. *Int J Hum Comput Interact*, 27(1):69–84, Jan 2011.

[25] M Salvaris and F Sepulveda. Visual modifications on the P300 speller BCI paradigm. *J Neural Eng*, 6(4):046011, Aug 2009.

[26] Gerwin Schalk, Dennis J McFarland, Thilo Hinterberger, Niels Birbaumer, and Jonathan R Wolpaw. BCI2000: a general-purpose brain-computer interface (BCI) system. *IEEE Trans Biomed Eng*, 51(6):1034–1043, Jun 2004.

[27] Thomas Schurmann and Peter Grassberger. Entropy estimation of symbol sequences. *Chaos*, 6(3):414–427, Sep 1996.

[28] Eric W. Sellers, Dean J. Krusienski, Dennis J. McFarland, Theresa M. Vaughan, and Jonathan R. Wolpaw. A P300 event-related potential brain–computer interface (BCI): The effects of matrix size and inter stimulus interval on performance. *Biological Psychology*, 73(3):242 – 252, 2006.

[29] C. E. Shannon. Prediction and entropy of printed english. *Bell Systems Technical Journal*, pages 50–64, 1951.

[30] N. K. Squires, K. C. Squires, and S. A. Hillyard. Two varieties of long-latency positive waves evoked by unpredictable auditory stimuli in man. *Electroencephalogr Clin Neurophysiol*, 38(4):387–401, Apr 1975.

[31] S Sutton, M Braren, J Zubin, and E R John. Evoked-potential correlates of stimulus uncertainty. *Science*, 150(3700):1187–1188, Nov 1965.

[32] G Townsend, B K LaPallo, C B Boulay, D J Krusienski, G E Frye, C K Hauser, N E Schwartz, T M Vaughan, J R Wolpaw, and E W Sellers. A novel P300-based brain-computer interface stimulus presentation paradigm: moving beyond rows and columns. *Clin Neurophysiol*, 121(7):1109–1120, Jul 2010.

[33] J.J. Vidal. Real-time detection of brain events in EEG. *Proceedings of the IEEE*, 65(5):633–641, May 1977.

[34] J R Wolpaw, N Birbaumer, W J Heetderks, D J McFarland, P H Peckham, G Schalk, E Donchin, L A Quatrano, C J Robinson, and T M Vaughan. Brain-computer interface technology: a review of the first international meeting. *IEEE Trans Rehabil Eng*, 8(2):164–73, Jun 2000.

[35] J R Wolpaw, D J McFarland, G W Neat, and C A Forneris. An EEG-based brain-computer interface for cursor control. *Electroencephalogr Clin Neurophysiol*, 78(3):252–259, Mar 1991.

[36] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, May 1977.

[37] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, Sep 1978.

A Experimental data

Subj	Sentence A PM 1	PM 2	P3S	Sentence B PM 1	PM 2
S1	6.48	8.94	2.51	4.16	8.94
S2	2.85	5.03	1.07	2.35	5.03
S3	4.11	5.65	1.06	2.64	5.65
S4	2.76	3.79	0.78	1.78	3.79
S5	3.87	5.32	0.88	2.49	3.80
S6	3.76	8.94	1.07	3.06	6.39
S7	3.94	6.92	1.45	3.23	6.92
S8	4.37	5.01	1.54	2.81	6.02
S9	3.66	4.28	0.93	2.35	5.03
S10	4.37	6.02	1.15	2.12	6.02

(a) Ouput characters per minute (OCM)

Subj	Sentence A PM 1	PM 2	P3S	Sentence B PM 1	PM 2
S1	1.97	1.94	2.51	1.99	1.94
S2	1.12	1.09	1.25	1.13	1.09
S3	1.25	1.23	1.43	1.26	1.23
S4	0.84	0.82	0.91	0.85	0.82
S5	1.18	1.16	1.34	1.19	1.16
S6	1.96	1.94	2.51	2.00	1.94
S7	1.54	1.50	1.83	1.54	1.50
S8	1.33	1.31	1.54	1.34	1.31
S9	1.11	1.12	1.25	1.13	1.09
S10	1.33	1.31	1.54	1.39	1.31

(b) Selections per minute (SM)

Table 4: Output characters per minute (OCM) and selections per minute (SM) for PolyMorph turn 1 (PM 1), PolyMorph turn 2 (PM 2), and P3Speller (P3S) for *in-vivo* experiments.

Subj	Sentence A PM 1	PM 2	P3S	Sentence B PM 1	PM 2
S1	11.71	12.00	12.00	11.55	12.00
S2	11.67	12.00	12.00	11.55	12.00
S3	11.71	12.00	12.00	11.55	12.00
S4	11.71	12.00	12.00	11.55	12.00
S5	11.71	12.00	12.00	11.55	12.00
S6	11.83	12.00	12.00	11.47	12.00
S7	11.56	12.00	12.00	11.55	12.00
S8	11.71	12.00	12.00	11.55	12.00
S9	11.71	11.67	12.00	11.55	12.00
S10	11.71	12.00	12.00	11.07	12.00

Table 5: Intensifications per selection and repetition (ISR) for PolyMorph turn 1 (PM 1), turn 2 (PM 2), and P3Speller (P3S) for *in-vivo* experiments.

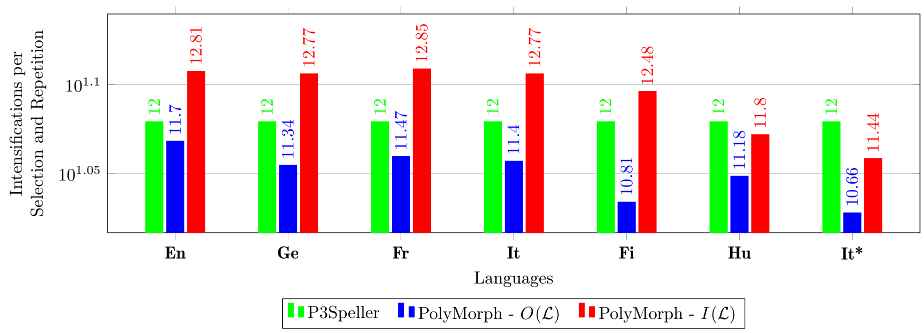


Figure 4: Intensifications per selection and repetition for *in-silico* experiments when predictions are enabled.

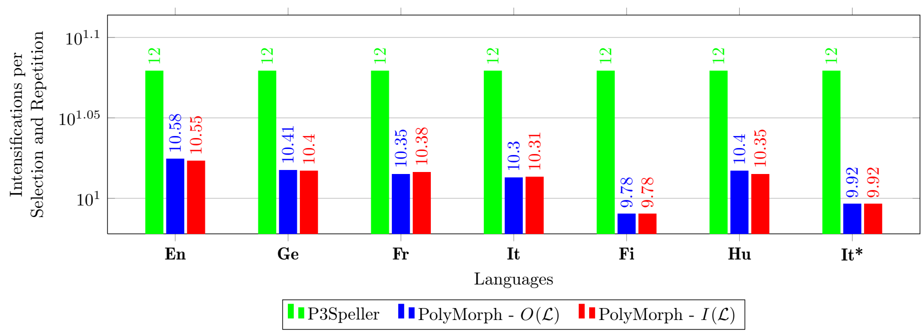


Figure 5: Intensifications per selection and repetition for *in-silico* experiments when predictions are disabled.

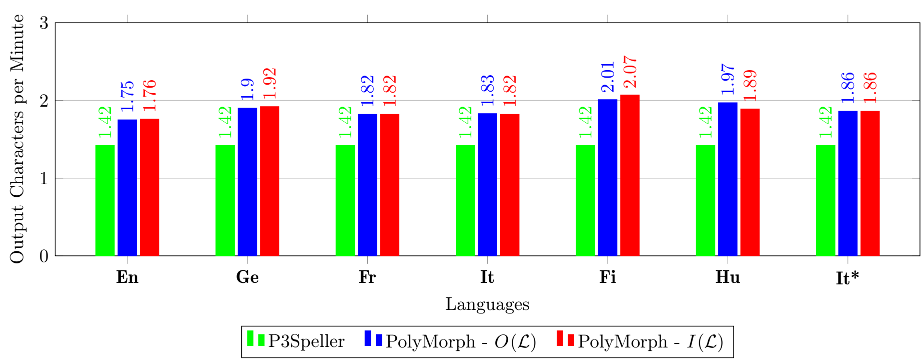


Figure 6: Mean output characters per minute for *in-silico* experiments when predictions are disabled.

	P3S		PM - Pred.			PM - No Pred.		
$\mathcal{L}$	Any	$\|h*w\|$	$O(\mathcal{L})$	$I(\mathcal{L})$	$\max\|h*w\|$	$O(\mathcal{L})$	$I(\mathcal{L})$	$\max\|h*w\|$
En	12.00	6×6	11.70	12.81	7×6	10.58	10.55	6×6
Ge	12.00	6×6	11.34	12.77	7×6	10.41	10.40	6×6
Fr	12.00	6×6	11.47	12.85	7×6	10.35	10.38	6×6
It	12.00	6×6	11.40	12.77	7×6	10.30	10.31	6×6
Fi	12.00	6×6	10.81	12.48	7×6	9.78	9.78	6×6
Hu	12.00	6×6	11.18	11.80	7×6	10.40	10.35	6×6
It*	12.00	6×6	10.66	11.44	6×6	9.92	9.92	6×5

Table 6: Intensifications per Selection and Repetition (ISR) for *in-silico* experiments. The presence of the prediction symbols in the selection matrix may increase the number of rows h. However, whenever the sentence to be spelt is not contained in the KB the ISR of PolyMorph is smaller than that of P3Speller. The data observed when we disabled predictions (i.e. PM - No Pred.) highlight the advantage of using PolyMorph over P3Speller in term of intensifications required to spell a sentence.

	P3S	PM - Pred.		PM - No Pred.	
$\mathcal{L}$	Any	$O(\mathcal{L})$	$I(\mathcal{L})$	$O(\mathcal{L})$	$I(\mathcal{L})$
En	1.43	2.66	6.08	1.75	1.76
Ge	1.43	2.84	7.05	1.9	1.92
Fr	1.43	2.8	6.41	1.82	1.82
It	1.43	2.66	6.36	1.83	1.82
Fi	1.43	2.74	7.55	2.01	2.07
Hu	1.43	3.62	6.23	1.97	1.89
It*	1.43	2.67	4.95	1.86	1.86

(a) Mean output characters per minute

	P3S	PM - Pred.		PM - No Pred.	
$\mathcal{L}$	Any	$O(\mathcal{L})$	$I(\mathcal{L})$	$O(\mathcal{L})$	$I(\mathcal{L})$
En	1.43	1.25	1.17	1.60	1.60
Ge	1.43	1.28	1.17	1.62	1.62
Fr	1.43	1.27	1.17	1.62	1.62
It	1.43	1.27	1.17	1.63	1.63
Fi	1.43	1.32	1.19	1.70	1.70
Hu	1.43	1.29	1.24	1.62	1.63
It*	1.43	1.34	1.27	1.68	1.68

(b) Mean selections per minute

Table 7: Mean output characters per minute (OCM) and selections per minute (SM) for *in-silico* experiments.